Campus de l'Université de Montréal
C.P. 6128, succursale Centre-ville
Montréal QC H3C 3J7 Canada

Telephone (514) 343-7575
Télécopieur (514) 343-7121

crt@crt.umontreal.ca
www.crt.umontreal.ca

CRT-2002-35

Centre de recherche sur les transports (C.R.T.)
Centre for Research on Transportation

HEURISTICS AND LOWER BOUNDS FOR THE BIN PACKING PROBLEM WITH CONFLICTS

by

Michel Gendreau
Gilbert Laporte
Frédéric Semet

October 2002

103-06369/

PUBLICATION
CRT-2002-35

HEURISTICS AND LOWER BOUNDS FOR THE BIN PACKING PROBLEM WITH CONFLICTS*

by

Michel Gendreau[1]
Gilbert Laporte[2]
Frédéric Semet[3]

* This research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

[1] Département d'informatique et de recherche opérationnelle and Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 <michelg@crt.umontreal.ca>

[2] Canada Research Chair in Distribution Management, HEC Montréal, 3000 chemin de la Côte-Sainte-Catherine, Montréal H3T 2A7 Canada and Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale Centre-Ville, Montréal H3C 3J7 Canada <gilbert@crt.umontreal.ca>

[3] LAMIH-ROI, Université de Valenciennes et du Hainaut Cambresis, Le Mont Houy, 59313 Valenciennes Cedex 9, France

Dépôt légal – Bibliothèque nationale du Québec
Bibliothèque nationale du Canada, 2002

Abstract

In the bin packing problem with conflicts, the aim is to pack items into the minimum number of bins subject to incompatibility restrictions. Two lower bounding procedures and six heuristics for this problem are developed and compared.

Key words: Bin packing problem, lower bounds, heuristics.

Résumé

Dans le problème de mise en boîtes avec conflits, on désire minimiser le nombre de boîtes dans lesquelles on veut placer un certain nombre d'items, dont certains sont incompatibles. On développe et compare deux bornes inférieures et six heuristiques pour ce problème.

Mots-clefs: problème de mise en boîtes, bornes inférieures, heuristiques.

1. Introduction

The purpose of this article is to describe and empirically evaluate heuristics and lower bounds for the *Bin Packing Problem with Conflicts* (BPC) defined as follows. We are given a set V of n items of weights $w_1, w_2, ..., w_n$, n identical *bins* of capacity c, with $w_1 \geq w_2 \geq ... \geq w_n$. Two items i and j are said to be *conflicting* if and only if they cannot be assigned to the same bin. The problem is to assign all items in the least possible number of bins while ensuring that the total weight of all items assigned to a bin does not exceed c and that no bin contains conflicting items. It is convenient to associate with the problem a *conflict graph* $G = (V, E)$, where an edge $(i, j) \in E$ exists if and only if items i and j conflict or if $w_i + w_j > c$. The BPC can be formulated as an integer linear program. Let y_k be a binary variable equal to 1 if and only if bin k is used, and let x_{ik} be a binary variable equal to 1 if and only if item i is assigned to bin k. The formulation is then

$$\text{minimize} \quad z = \sum_{k=1}^{n} y_k \tag{1}$$

subject to

$$\sum_{i=1}^{n} w_i x_{ik} \leq c y_k \qquad (k = 1, ..., n) \tag{2}$$

$$\sum_{k=1}^{n} x_{ik} = 1 \qquad (i = 1, ..., n) \tag{3}$$

$$x_{ik} + x_{jk} \leq 1 \quad ((i, j) \in E, k = 1, ..., n) \tag{4}$$

$$y_k = 0 \text{ or } 1 \quad (k = 1, ..., n) \tag{5}$$

$$x_{ik} = 0 \text{ or } 1 \qquad (i, k = 1, ..., n). \tag{6}$$

A well-known application of the BPC arises in examination scheduling. A common version of this problem (see, e.g., Laporte and Desroches, 1984) is where n examinations with enrolments w_i must be scheduled in the least possible number of time periods. Any two examinations with students in common cannot be scheduled in the same period, and the total seating capacity in each period is

equal to c. Another application quoted by Jansen (1999) is the assignment of processes to processors where some processes cannot be executed to the same processor for reason of fault tolerance or for efficiency purposes. Jansen (1999) also mentions an application found in load balancing of tasks in parallel computing where some subcomputations cannot be performed simultaneously. The BCP is also encountered in delivery problems where some flammable, explosive or toxic substances cannot be placed in the same vehicle (Christofides, Mingozzi and Toth, 1979).

The BPC is strongly NP-hard since it generalizes the *Bin Packing Problem* (BPP) in which all items are non-conflicting, see, e.g., Martello and Toth (1990a). It also generalizes the *Bounded Vertex Coloring Problem* (BVCP) in which $w_i = 1$ for each $v_i \in V$ (see, e.g., Hansen, Hertz and Kuplinsky, 1993). For a typology and additional references on bin packing problems, see Dyckhoff (1990).

The BCP has been studied by Jansen and Oehring (1997) and by Jensen (1999) who have shown that no bounded polynomial approximation scheme can be derived for general conflict graphs. These authors have derived polynomial approximation schemes with bounded worst-case performance ratios for particular classes of conflict graphs (so called p-inductive graphs in which each vertex is assigned a distinct number in $\{1, ..., n\}$ and each vertex is adjacent to at most p lower numbered vertices). Similar results have also been derived by Baker and Coffman (1996) for the BVCP. Bodlaender and Jansen (1993) have analyzed the complexity of the BVCP for different conflict graph classes. Carter, Laporte and Lee (1996) have compared a number of greedy assignment heuristics for scheduling a given number of conflicting examinations in the least possible number of periods, under capacity restrictions.

In this article we develop six heuristics for the BPC, and we test them on several benchmark instances. No assumptions are made on the adjacency structure of the graph. Lower bounds are developed to help assess the value of the heuristics. We describe the heuristics in Section 2 and the lower bounds in Section 3. Computational results are presented in Section 4, followed by a conclusion in Section 5.

2. Heuristics

We now describe six heuristics for the BPC. The first is a simple adaptation of a well-known heuristic for the BPP developed by Coffman, Garey and Johnson (1984). The next three make use of a graph colouring procedure (see, e.g., Brélaz, 1979). The last two are based on the computation of a clique of non-conflicting or of conflicting items (see, e.g., Johnson, 1974).

2.1 First fit decreasing packing heuristics

The following algorithm is a direct adaptation of the first fit decreasing (FFD) packing algorithm of Coffman, Garey and Johnson (1984) since the items are already sorted, with an additional test for the absence of conflicts.

H1. Modified FFD

Consider each item in turn and assign it to the first bin in which there is sufficient residual capacity and for which there is no conflict with the already assigned items.

When all items are mutually non-conflicting, the complexity of the FFD heuristic is $O(n \log n)$ (Coffman, Garey and Johnson, 1984). However, since $O(n^2)$ conflicts are present, the complexity of H1 is also $O(n^2)$.

2.2 Heuristics based on graph colouring

The following three algorithms make use of the conflict graph G. A colouring of the vertices of V is first performed by means of the DSatur (or degree saturation) heuristic proposed by Brélaz (1979). This heuristic maintains a partition $\{C, U\}$ of V where C is the set of coloured vertices and U is the set of uncolored vertices. It can be described as follows.

Step 1. Set $C := \varnothing$ and $U := V$. Arrange the vertices by decreasing order of degrees.

Step 2. Identify a vertex $v \in U$ of maximal degree and assign colour 1 to v. Set $C := C \cup \{v\}$ and $U := U \setminus \{v\}$.

Step 3. Choose a vertex $v \in U$ with maximal *saturation degree*, i.e., the maximum number of colours to which it is adjacent among the vertices of C. In case of equality, choose the vertex of maximal degree.

Step 4. Assign to v the least possible colour number. Set $C := C \cup \{v\}$ and $U := U \setminus \{v\}$.

Step 5. If $C = V$, stop. Otherwise, go to Step 3.

This algorithm can be executed in $O(n^2 \log n)$ time.

Heuristic H2 which is similar to that of Jansen and Oehring (1997) uses a colouring procedure to partition the items into sets of mutually non-conflicting items, and it then solves a BPP for each set.

H2. Colouring and FFD

Step 1. Colour the vertices of V using DSatur.

Step 2. Solve a separate BPP for the items of each colour using FFD.

Heuristic H3 is similar to H2, except that it initially removes the items with zero or only one conflict. These items are then assigned to bins using a modified FFD heuristic.

H3. Partial colouring and FFD.

Step 1. Remove from V the subset Z of vertices having a degree of 0 or 1 in G.

Step 2. Apply H2 to the items of $V \setminus Z$.

Step 3. Assign the items of Z to the bins just created, and possibly to new bins by means of H1.

The complexity of H2 and H3 is $O(n^2 \log n)$ since it is dominated by the colouring procedure.

The last heuristic of this category first identifies a large subset of mutually non-conflicting items, and packs them into bins. All bins whose residual capacity exceeds a certain treshold, except the most filled bin, are emptied and the process is repeated with all items yet unassigned. Convergence is guaranteed since at least one bin is not emptied in Step 3.

H4. Coloring and iterated FFD.

Step 1. Colour the vertices of V using DSatur.

Step 2. Extract from V a largest subset P of vertices having the same colour. If $|P| \neq 1$, go to Step 3. Otherwise assign each item of V to a different bin and stop.

Step 3. Solve a BPP on P using FFD. Identify the item $h \in P$ having the smallest weight and define the set Q of all items belonging to bins whose residual capacity is larger than w_h, except those belonging to the most filled bin.

Step 4. Set $V := (V \setminus P) \cup Q$. If $V \neq \emptyset$, go to Step 1.

The complexity of Step 1 is again $O(n^2 \log n)$. Step 3 has a complexity of $O(n \log n)$ and is applied at most n^2 times. The overall complexity of H4 is therefore $O(n^3 \log n)$.

2.3 Heuristics based on clique computations

The last two heuristics compute a large clique in the non-conflict graph $\overline{G} = (V, \overline{E})$ or in the conflict graph $G = (V, E)$. In $\overline{G}$, an edge $(i,j) \in \overline{E}$ exists if and only if i and j are non-conflicting and $w_i + w_j \leq c$. The cliques are determined by means of Johnson's (1974) greedy heuristic. This algorithm maintains a partition $\{C, U\}$ of V where C is the set of vertices included in the clique and U is the set of vertices not yet included in the clique. Initially C contains a vertex of maximal degree. At each step, the procedure transfers to C a vertex of U linked to all vertices of C and having a maximum degree. This algorithm can be implemented in $O(n \log n)$ time.

The next heuristic identifies a clique of non-conflicting items and assigns them to bins. As for heuristic H4, some bins with a large residual capacity are emptied and the process is repeated.

H5. Clique in $\overline{G}$ and FFD

Step 1. Determine a large clique D in $\overline{G}$ using Johnson's algorithm. If $|D| \neq 1$, go to Step 2. Otherwise assign each item of V to a different bin and stop.

Step 2. Solve a BPP on the clique D using FFD. Identify item $h \in D$ having the smallest weight and define the set Q of all items belonging to bins whose residual capacity is larger than w_h, except those belonging to the most filled bin.

Step 3. Set $V := (V \setminus D) \cup Q$. If $V \neq \emptyset$, go to Step 1.

The complexity of Steps 1 and 2 is $O(n \log n)$. Since each of these steps can be executed at most n times, the overall complexity of H5 is $O(n^2 \log n)$.

The last heuristic first removes the items that are non-conflicting with all other items. It then determines a large clique D of conflicting items. Each item i of D is then considered to compute a clique D_i of non-conflicting items. Then, by solving a BPP, we identify a subset of D_i which can be packed with i in the same bin.

H6. Clique on G and $\overline{G}$, and FFD.

Step 1. Determine the set H of items non-conflicting with all items of V, and set $V := V \setminus H$.

Step 2. Using Johnson's heuristic, determine a clique D in the conflict graph $G = (V, E)$, and set $V := V \setminus D$. If $V = \varnothing$, stop.

Step 3. For every $i \in D$ compute a large clique D_i containing i in the non-conflict graph $\overline{G}_i$ induced by $V \cup \{i\}$ using Johnson's heuristic. Use FFD to solve the BPP on D_i. Remove from V all items included in the bin containing i and set $D := D \setminus \{i\}$. If $V = \varnothing$ go to Step 4. If $D = \varnothing$, go to Step 2. Otherwise, go to Step 3.

Step 4. Assign the items of H to the existing bins, and possibly new ones, using FFD.

The complexity of Step 1 is $O(n^2)$, that of Step 2 is $O(n \log n)$ and Step 3, which is applied at most n times, has a complexity of $O(n \log n)$ for each i. Therefore the overall complexity of H6 is $O(n^2 \log n)$.

3. Lower bounds

We have developed two lower bounds in order to better assess the performance of the heuristics developed in Section 2. The first is solely based on bin packing considerations while the second also takes conflict constraints into account.

Bin packing lower bound L1

A valid lower bound can be obtained by solving a BPP optimally, disregarding conflict constraints. Since the BPP is NP-hard, we use instead the simple lower bound L1$= \left\lceil \sum_{i=1}^{n} w_i/c \right\rceil$ on its solutions value.

Constrained packing lower bound L2

The constrained packing lower bound first assigns to different bins a set D of mutually conflicting items. Then, for a given weight level ℓ, it considers the set I_ℓ of non-assigned items whose weight is at least ℓ, but less than the residual capacity of at least one bin. It then considers K_ℓ, the set of bins with sufficient residual capacity to accomodate the items of I_ℓ. Then, one can compute a lower bound on the total weight z_ℓ of items of I_ℓ that cannot fit in any of the existing bins. This is done by solving a *Transportation Problem* (TP) with "origin set" I_ℓ, "destination set" K_ℓ, and suitably defined transportation costs. Here is a step by step description of the second lower bounding procedure.

Step 1. Determine a large clique D on $G = (V, E)$ using Johnson's (1974) heuristic, and assign each item of D to a different bin. Let r_k be the residual capacity of bin k and let $h = \underset{i \in D}{\operatorname{argmin}} \{w_i\}$.

Step 2. For a given weight level $\ell \leq c - w_h$, define the three sets

$$\begin{aligned} I'_\ell &= \{i \in V : w_i < \ell\}, \\ I_\ell &= \{i \in V : \ell \leq w_i \leq c - w_h\}, \\ I''_\ell &= \{i \in V : \ell > c - w_h\}. \end{aligned}$$

In addition to the already created $|D|$ bins, create a dummy bin 0 of infinite residual capacity. Let K_ℓ be the set of bins k having a residual capacity $r_k \geq \ell$.

Step 3. Define the following TP consisting of optimally assigning entire or partial items $i \in I_\ell$ to bins $k \in K_\ell$, with the following costs c_{ik}. For $k \geq 1$, c_{ik} is equal to 1 if i is non-conflicting with the item already assigned to k, and to ∞ otherwise. For $k = 0, c_{i0} = n$ for all $i \in I_\ell$. Let x_{ik} be a variable equal to the portion of the weight of item i assigned to bin k. The TP is then

$$\text{minimize} \qquad \sum_{i \in I_\ell} \sum_{k \in K_\ell} c_{ik} x_{ik} \tag{7}$$

subject to

$$\sum_{k \in K_\ell} x_{ik} = w_i \qquad (i \in I_\ell) \tag{8}$$

$$\sum_{i \in I_\ell} x_{ik} \leq r_k \qquad (k \in K_\ell) \tag{9}$$

$$x_{ik} \quad \text{integer} \qquad (i \in I_\ell, k \in K_\ell). \tag{10}$$

For the weight level ℓ considered, a valid BPC lower bound is given by

$$b_\ell = |D| + \left\lceil \left(\sum_{i \in I_\ell} x_{i0} + \sum_{i \in I''_\ell} w_i + z_\ell \right) / c \right\rceil. \tag{11}$$

In (11), the expression in parentheses is the sum of 1) the weight of I_ℓ that cannot feasibly fit in the $|D|$ bins identified in Step 1, 2) all weight of I''_ℓ since none of the items of that set can fit into any of the initial $|D|$ bins, and 3) a lower bound z_ℓ on the total weight of the items of I_ℓ that cannot fit into any of the initial $|D|$ bins. Formally, z_ℓ is computed as follows:

$$z_\ell = \max \left\{ 0, \sum_{i \in I'_\ell} w_i - \left(\sum_{k \in D} r_k - \sum_{i \in I_\ell} \sum_{k \in K_\ell \setminus \{0\}} x_{ik} \right) \right\}. \tag{12}$$

In (12) the term in parentheses computes the residual capacity of all bins of D once parts of the items of I_ℓ have been assigned to them. If this residual capacity exceeds the total weight of all items of I'_ℓ, then y_ℓ is 0; otherwise y_ℓ is the portion of that weight over the total residual capacity.

Since b_ℓ is a valid BPC lower bound for any $\ell \in W = \{w_i; w_i \leq c - w_h\}$, it follows that

$$\max_{\ell \in W} \{b_\ell\}. \tag{13}$$

is also a valid BPC lower bound.

4. Computational results

The algorithms just described were coded in C. The transportation algorithm of Section 3 was translated from the Pascal code of Syslo, Deo and Kowalik (1983). All programs were run of a Sun Ultrasparc Station with a 140 MHz processor.

Several instances of the BPC were constructed as follows. We first started with some of the uniform bin packing instances generated by Falkenauer (1996) which are widely recognized as being rather difficult. More specifically we took the first ten uniform instances for each value of $n = 120, 250, 500,$ and 1000. They consist of item weights discretely, and uniformly distributed in [20, 100] and of bin capacities c =150. For the second batch of tests, we worked with the triplet bin packing instances of Falkenauer (1996) generated in such a way that in the optimal BPP solution, each bin contains exactly three items. We took the first ten instances for each value of n = 60, 120, 249, and 501 and multiplied the item weights by 10 to obtain integer data. The bin capacity was set equal to c =1000 in all cases. The conflict graph G was generated by using the procedure described in Soriano and Gendreau (1996). A value p_i was first assigned to each vertex $i \in V$ according to a continuous uniform distribution on [0,1]. Each edge (i, j) of G was created whenever $(p_i + p_j)/2 \leq d$, where d is the expected density of G. We used d =0%, 10%, ... , 90%, yielding 800 instances in total.

A time limit of 1200 CPU seconds was imposed for the solution of any instance. In all cases, except for heuristic H4 and some large values of n, this limit was never reached. Tables 1 to 8 contain average results over 10 instances. Best result are shown in boldfaced characters. The table headings are as follows:

d : expected density of the conflict graph;

Li : value of the i^{th} lower bound;

UB/LB : upper to lower bound ratio for each heuristic where $LB = \max\{L1, L2\}$;

sec : computation time in seconds;

nb opt. : number of proven optima out of 10 instances (for H6 only).

Computational results indicate that the lower bound L2 is identical to L1 for small densities (typically $d \leq$20%), but becomes vastly superior to L1 when d increases. This is to be expected since L1 does not take conflicts into account whereas L2 does. The best heuristic is usually H6, followed by H1. The latter heuristic tends to be marginally better than H6 when $d = 20\%$ or 30% and is always

much faster. Our results show that the Falkenauer triplet instances are particularly difficult for pure bin packing problems ($d = 0\%$), but relatively easy otherwise. When $d = 0\%$, the optimal solution always contains $n/3$ bins and the average UB/LB ratio is always at least equal to 1.138, even for H1 which is known to systematically produce near optimal solutions on uniform instances. Finally, for both classes of instances, even the best heuristic (H6) rarely produces proven optimal solutions, it tends to be more successful in this respect when the density becomes larger.

Table 1 Computational results for the Falkenauer uniform instances (n = 120, c = 150)

d	L1	L2	MAX (L1,L2)	H1		H2		H3		H4		H5		H6		
				UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	nb. opt.	sec.
0%	48,3	48,3	48,3	**1,017**	0,1	1,279	0,5	1,049	0,5	1,261	1,9	1,302	1,3	**1,017**	2	0,2
10%	48,3	48,3	48,3	1,025	0,1	1,281	0,6	1,080	0,4	1,242	2,1	1,263	1,2	**1,017**	2	0,2
20%	48,3	48,3	48,3	**1,056**	0,1	1,291	0,6	1,109	0,6	1,180	2,8	1,213	1,2	**1,056**	0	0,3
30%	48,3	48,3	48,3	**1,152**	0,2	1,295	0,7	1,192	0,7	1,188	4,6	1,237	1,0	1,192	0	0,3
40%	48,3	49,7	49,7	1,248	0,2	1,351	1,0	1,289	1,0	1,289	13	1,349	0,8	**1,182**	0	0,4
50%	48,3	59,8	59,8	1,197	0,2	1,265	1,3	1,260	1,3	1,242	15	1,284	0,6	**1,108**	0	0,5
60%	48,3	72,1	72,1	1,130	0,2	1,178	1,7	1,178	1,8	1,165	13,5	1,192	0,4	**1,091**	0	0,5
70%	48,3	85,8	85,8	1,081	0,3	1,100	2,0	1,100	2,0	1,091	16,4	1,112	0,3	**1,056**	0	0,7
80%	48,3	96,3	96,3	1,046	0,4	1,059	2,2	1,059	2,3	1,051	15,7	1,063	0,4	**1,033**	0	0,6
90%	48,3	108,3	108,3	1,017	0,4	1,029	2,4	1,029	2,4	1,017	16,1	1,023	0,3	**1,015**	4	0,6

Table 2 Computational results for the Falkenauer uniform instances (n = 250, c = 150)

d	L1	L2	MAX (L1,L2)	H1		H2		H3		H4		H5		H6		
				UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	nb. opt.	sec.
0%	101,4	101,4	101,4	**1,015**	1,5	1,327	3,0	1,098	3,0	1,299	14,4	1,327	6,6	**1,015**	0	0,8
10%	101,4	101,4	101,4	1,020	0,9	1,323	2,7	1,135	2,8	1,277	14,3	1,278	6,1	**1,016**	0	1,2
20%	101,4	101,4	101,4	**1,047**	0,9	1,278	3,2	1,135	3,1	1,211	21	1,230	5,6	1,051	0	2,1
30%	101,4	101,4	101,4	**1,107**	0,9	1,258	3,6	1,163	3,6	1,184	32,9	1,216	4,9	1,226	0	2,7
40%	101,4	102,0	102,0	1,246	0,8	1,332	5,1	1,284	4,4	1,305	55	1,352	4,2	**1,179**	0	3,4
50%	101,4	121,6	121,6	1,204	0,9	1,259	6,0	1,256	6,2	1,237	87,6	1,281	2,7	**1,126**	0	3,8
60%	101,4	149,6	149,6	1,127	1,0	1,164	7,5	1,164	7,6	1,155	115,2	1,179	2,1	**1,088**	0	3,7
70%	101,4	177,8	177,8	1,068	1,2	1,098	8,8	1,098	8,8	1,093	129,7	1,104	1,3	**1,030**	0	3,9
80%	101,4	199,8	199,8	1,049	1,4	1,062	9,9	1,062	9,9	1,055	121,4	1,068	1,1	**1,030**	0	10,3
90%	101,4	225,0	225,0	1,019	1,6	1,029	10,6	1,029	10,6	1,024	105	1,029	0,9	**1,010**	1	4,4

Table 3 Computational results for the Falkenauer uniform instances (n = 500, c = 150)

d	L1	L2	MAX (L1,L2)	H1		H2		H3		H4		H5		H6		
				UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	nb. opt.	sec.
0%	202,6	202,6	202,6	**1,013**	0,9	1,356	18,0	1,128	17,8	1,331	130,3	1,331	47,0	**1,013**	0	5,4
10%	202,6	202,6	202,6	1,017	0,9	1,324	19,1	1,140	18,8	1,292	164,3	1,282	44,1	**1,015**	0	9,3
20%	202,6	202,6	202,6	**1,043**	1,0	1,280	22,2	1,142	21,7	1,219	291,7	1,229	39,6	1,053	0	16,3
30%	202,6	202,6	202,6	**1,103**	1,0	1,245	28,0	1,152	27,5	1,186	475	1,230	34,3	1,202	0	29,7
40%	202,6	205,2	205,2	1,231	1,6	1,348	35,2	1,302	35,1	1,319	771,6	1,359	26,0	**1,186**	0	32,3
50%	202,6	252,2	252,2	1,169	3,0	1,244	48,7	1,244	46,2	- -	- -	1,256	17,2	**1,085**	0	35,8
60%	202,6	302,2	302,2	1,108	4,3	1,162	60,6	1,162	58,2	- -	- -	1,173	12,1	**1,050**	0	30,3
70%	202,6	348,0	348,0	1,082	5,8	1,113	67,5	1,113	68,0	- -	- -	1,114	7,2	**1,040**	0	30,1
80%	202,6	397,7	397,7	1,050	7,4	1,067	75,5	1,067	76,0	- -	- -	1,072	4,5	**1,032**	0	31,5
90%	202,6	452,0	452,0	1,022	9,2	1,027	83,5	1,027	81,9	- -	- -	1,029	3,4	**1,013**	0	32,5

Table 4 Computational results for the Falkenauer uniform instances (n = 1000, c = 150)

d	L1	L2	MAX (L1,L2)	H1		H2		H3		H4		H5		H6		
				UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	nb. opt.	sec.
0%	401,8	401,8	401,8	**1,012**	0,9	1,372	134,3	1,141	134,6	- -	- -	1,332	374,2	**1,012**	0	40,3
10%	401,8	401,8	401,8	1,014	1,1	1,333	142,5	1,146	142,8	- -	- -	1,277	346,1	**1,012**	0	83,1
20%	401,8	401,8	401,8	**1,037**	1,8	1,255	167,2	1,137	167,6	- -	- -	1,226	303,3	1,058	0	336,3
30%	401,8	401,8	401,8	**1,095**	3,2	1,233	211,2	1,139	211,1	- -	- -	1,240	251,2	1,216	0	670,4
40%	401,8	407,5	407,5	1,217	7,9	1,363	275,1	1,318	274,8	- -	- -	1,375	193,6	**1,149**	0	809
50%	401,8	509,5	509,5	1,138	16,2	1,238	364,2	1,237	363,6	- -	- -	1,251	140,2	**1,063**	0	810,9
60%	401,8	602,3	602,3	1,098	25,6	1,161	498,3	1,161	459,1	- -	- -	1,173	80,1	**1,047**	0	575,4
70%	401,8	707,4	707,4	1,063	37,0	1,103	539,2	1,103	538,5	- -	- -	1,108	44,4	**1,027**	0	515,3
80%	401,8	799,2	799,2	1,042	49,4	1,065	598,9	1,065	597,2	- -	- -	1,066	24,7	**1,024**	0	512,2
90%	401,8	899,2	899,2	1,020	63,0	1,028	649,9	1,028	642,2	- -	- -	1,031	15,4	**1,011**	0	501,0

Table 5 Computational results for the Falkenauer triplet instances ($n = 60, c = 1000$)

			MAX	H1		H2		H3		H4		H5		H6		
d	L1	L2	(L1,L2)	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	nb. opt.	sec.
0%	20,0	20,0	20,0	**1,155**	0,1	**1,155**	0,1	**1,155**	0,1	**1,155**	0,2	**1,150**	0,1	**1,150**	0	0,1
10%	20,0	20,0	20,0	1,150	0,1	1,295	0,1	**1,150**	0,1	1,255	0,1	1,260	0,1	**1,150**	0	0,1
20%	20,0	20,0	20,0	1,200	0,1	1,530	0,1	1,140	0,1	1,495	0,1	1,505	0,2	**1,130**	0	0,1
30%	20,0	20,7	20,7	1,217	0,1	1,626	0,1	1,145	0,1	1,596	0,1	1,631	0,2	**1,109**	1	0,1
40%	20,0	24,2	24,2	1,159	0,1	1,531	0,1	1,291	0,1	1,493	0,2	1,517	0,2	**1,149**	2	0,1
50%	20,0	29,3	29,3	1,121	0,1	1,370	0,1	1,331	0,2	1,329	0,3	1,356	0,1	**1,053**	1	0,2
60%	20,0	35,2	35,2	1,091	0,1	1,254	0,1	1,254	0,1	1,237	0,1	1,243	0,1	**1,053**	2	0,1
70%	20,0	41,9	41,9	1,046	0,1	1,150	0,1	1,150	0,1	1,141	0,1	1,148	0,1	**1,035**	3	0,1
80%	20,0	47,7	47,7	1,035	0,1	1,083	0,1	1,083	0,1	1,073	0,2	1,084	0,1	**1,033**	1	0,1
90%	20,0	53,9	53,9	**1,011**	0,1	1,036	0,1	1,036	0,1	1,027	0,2	1,029	0,1	**1,011**	5	0,1

Table 6 Computational results for the Falkenauer triplet instances ($n = 120, c = 1000$)

			MAX	H1		H2		H3		H4		H5		H6		
d	L1	L2	(L1,L2)	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	nb. opt.	sec.
0%	40,0	40,0	40,0	**1,138**	0,1	**1,138**	0,2	**1,138**	0,3	**1,138**	0,3	**1,138**	0,9	**1,139**	0	0,1
10%	40,0	40,0	40,0	**1,138**	0,1	1,333	0,2	1,150	0,3	1,308	0,2	1,320	1,0	1,145	0	0,1
20%	40,0	40,0	40,0	1,158	0,1	1,535	0,3	**1,125**	0,3	1,513	0,3	1,525	0,9	1,130	0	0,2
30%	40,0	40,6	40,6	1,232	0,1	1,695	0,4	1,251	0,3	1,680	0,5	1,692	0,8	**1,100**	0	0,4
40%	40,0	50,4	50,4	1,144	0,1	1,509	0,4	1,315	0,5	1,495	0,7	1,499	0,7	**1,045**	0	0,7
50%	40,0	61,8	61,8	1,072	0,2	1,345	0,6	1,329	0,7	1,325	1,5	1,344	0,5	**1,040**	0	1,4
60%	40,0	72,9	72,9	1,054	0,2	1,238	0,9	1,238	0,9	1,213	2,5	1,233	0,4	**1,031**	1	2,2
70%	40,0	83,0	83,0	1,048	0,2	1,164	1,1	1,164	1,1	1,148	2,7	1,159	0,3	**1,030**	3	1,1
80%	40,0	96,8	96,8	1,027	0,3	1,083	1,2	1,083	1,2	1,078	3,1	1,083	0,3	**1,016**	2	1,0
90%	40,0	107,6	107,6	1,018	0,3	1,039	1,3	1,039	1,3	1,033	3,5	1,037	0,3	**1,013**	1	1,1

Table 7 Computational results for the Falkenauer triplet instances $(n = 249, c = 1000)$

			MAX	H1		H2		H3		H4		H5		H6		
d	L1	L2	(L1,L2)	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	nb. opt.	sec.
0%	83,0	83,0	83,0	**1,143**	0,2	**1,143**	1,7	**1,143**	1,6	**1,143**	1,5	**1,143**	7,8	**1,143**	0	0,1
10%	83,0	83,0	83,0	**1,143**	0,1	1,328	1,7	**1,143**	1,6	1,313	1,6	1,320	7,5	1,145	0	0,3
20%	83,0	83,0	83,0	1,151	0,1	1,530	1,9	1,130	1,9	1,514	2,3	1,524	7,0	**1,128**	0	1,2.
30%	83,0	83,2	83,2	1,230	0,3	1,703	2,5	1,247	2,4	1,689	3,9	1,696	5,9	**1,117**	0	3,2
40%	83,0	106,1	106,1	1,120	0,5	1,507	3,5	1,331	3,6	1,493	8,5	1,507	4,4	**1,027**	0	8,0
50%	83,0	125,9	125,9	1,079	0,7	1,371	5,0	1,366	5,0	1,358	13,5	1,366	3,4	**1,028**	0	15,6
60%	83,0	150,5	150,5	1,058	0,9	1,247	6,8	1,247	6,9	1,241	17,2	1,247	2,2	**1,024**	0	11,5
70%	83,0	175,5	175,5	1,035	1,2	1,158	8,4	1,158	8,3	1,152	25	1,156	1,5	**1,020**	0	9,0
80%	83,0	199,1	199,1	1,027	1,4	1,093	9,5	1,093	9,6	1,088	29,2	1,091	1,0	**1,015**	0	8,3
90%	83,0	226,3	226,3	1,011	1,7	1,035	10,3	1,035	10,4	1,032	34,5	1,034	0,7	**1,006**	1	8,4

Table 8 Computational results for the Falkenauer triplet instances $(n = 501, c = 1000)$

			MAX	H1		H2		H3		H4		H5		H6		
d	L1	L2	(L1,L2)	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	sec.	UB/LB	nb. opt.	sec.
0%	167,0	167,0	167,0	**1,140**	0,3	**1,140**	11,3	**1,140**	11,1	**1,140**	11,1	**1,140**	61,3	**1,140**	0	0,4
10%	167,0	167,0	167,0	1,141	0,3	1,333	11,3	**1,138**	11,4	1,323	12,1	1,329	61,0	**1,138**	0	2,0
20%	167,0	167,0	167,0	1,143	0,5	1,526	13,5	1,122	13,1	1,520	16,8	1,523	54,8	**1,121**	0	11,0
30%	167,0	169,0	169,0	1,192	0,9	1,703	17,6	1,254	17,4	1,688	40,5	1,699	47,0	**1,102**	0	41,0
40%	167,0	198,8	198,8	1,119	1,7	1,576	25,4	1,375	24,7	1,567	65,4	1,574	37,4	**1,025**	0	118,5
50%	167,0	125,9	254,8	1,047	3,1	1,365	38,6	1,363	38,2	1,358	114,6	1,358	25,2	**1,016**	1	227,5
60%	167,0	298,6	298,6	1,043	4,7	1,256	53,1	1,256	52,6	1,249	173,9	1,254	19,3	**1,019**	0	131,5
70%	167,0	356,1	356,1	1,030	6,7	1,153	66,7	1,153	66,2	1,150	221,6	1,152	8,3	**1,012**	1	81,8
80%	167,0	402,8	402,8	1,019	8,1	1,091	75,6	1,091	75,5	1,088	245,6	1,090	4,9	**1,010**	2	67,1
90%	83	450,6	450,6	1,011	9,8	1,040	81,8	1,040	82,3	1,039	302,8	1,041	3,4	**1,006**	2	65,1

5. Conclusion

We have developed, tested and compared several heuristics for the Bin Packing Problem with Conflicts, a hard combinatorial optimization problem. The best of these heuristics, H6, combines a clique computation and a first fit decreasing procedure. We have also described a new lower bound that exploits the conflict graph of the problem and is vastly superior to the standard bin packing bound.

Acknowledgements

This work was partially supported by the Canadian Natural Sciences and Engineering Research Council under grants OGP0038816, OGP0039682 and OGP0184123. This support is gratefully acknowledged. Thanks are due to Frédéric Flament for his help with programming, and to the referees for their valuable comments.

References

Baker, B.S., and E.G. Coffman (1996). "Mutual Exclusion Scheduling", *Theoretical Computer Science* 162, 225–243.

Bodlaender, H.-L., and K. Jansen (1993). "On the Complexity of Scheduling Incompatible Jobs with Unit-Times", *Mathematical Foundations of Computer Science*, MFCS 93, LNCS 711, 291–300.

Brélaz, D. (1979). "New Methods to Color the Vertices of a Graph", *Communications of the Association for Computing Machinery* 22, 251–256.

Carter, M.W., G. Laporte, and S.Y. Lee (1996). "Examination Timetabling: Algorithmic Strategies and Applications", *Journal of the Operational Research Society* 47, 373–383.

Christofides, N., A. Mingozzi, and P. Toth (1979). "Loading Problems". In N. Christofides, A. Mingozzi, P. Toth and C. Sandi (eds.), *Combinatorial Optimization*, pp. 339–369, Chichester: Wiley.

Coffman, E.G. Jr., M.R. Garey, and D.S. Johnson (1984). "Approximation Algorithms for Bin-Packing – An Updated Survey". In G. Ausiello, M. Lucertini and P. Serafini (eds.), *Algorithm Design for Computer System Design*, pp. 49–106, Vienna: Springer.

Dycknoff, H. (1990), "A Typology of Cutting and Packing Problems", *European Journal of Operational Research* 44, 145–159.

Falkenauer, E. (1996), "A Hybrid Grouping Genetic Algorithm", *Journal of Heuristics* 2, 5–30.

Hansen, P., A. Hertz, and J. Kuplinsky (1993). "Bounded Vertex Colorings of Graphs", *Discrete Mathematics* 111, 305–312.

Jansen, K. (1999). "An Approximation Scheme for Bin Packing with Conflicts", *Journal of Combinatorial Optimization* 3, 363–377.

Jansen, K., and S. Oehring (1997). "Approximation Algorithms for Time Constrained Scheduling", *Information and Computation* 132, 85-108.

Johnson, D.S. (1974). "Approximation Algorithms for Combinatorial Problems", *Journal of Computer and System Science* 9, 256–278.

Laporte, G., and S. Desroches (1984). "Examination Timetabling by Computer", *Computers & Operations Research* 11, 351–360.

Martello, S., and P. Toth (1990a). *Knapsack Problems*. Chichester: Wiley.

Martello, S., and P. Toth (1990b). "Lower Bounds and Reduction Procedures for the Bin Packing Problem", *Discrete Applied Mathematics* 28, 59–70.

Soriano, P., and M. Gendreau (1996). "Tabu Search Algorithms for the Maximum Clique Problem". In D.S. Johnson and M.A. Trick (eds.), *Cliques, Coloring and Satisfiability*, pp. 221–242. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 26, Providence: American Mathematical Society.

Syslo, M.M., N. Deo, and J.S. Kowalik (1983). *Discrete Optimization Algorithms with* PASCAL *Programs*, Englewood Cliffs: Prentice-Hall.